UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/761,404	01/16/2001	Bjarne Steensgaard	MS 158288.1/40062.119US01	4797

23552 7590 02/09/2005

MERCHANT & GOULD PC
P.O. BOX 2903
MINNEAPOLIS, MN 55402-0903

EXAMINER	
ALI, SYED J	

ART UNIT	PAPER NUMBER
2127	

DATE MAILED: 02/09/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/761,404	STEENSGAARD, BJARNE	
	Examiner	Art Unit	
	Syed J Ali	2127	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 November 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-51* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-51* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on October 20, 2004 has been entered.

2. This office action is in response to the amendment filed October 20, 2004. Claims 1-51 are presented for examination.

3. The text of those sections of Title 35, U.S. code not included in this office action can be found in a prior office action.

Claim Rejections - 35 USC § 103

4. **Claims 1-6, 15-26, 31-32, and 34-51 rejected under 35 U.S.C. 103(a) as being unpatentable over Jagannathan et al. (USPN 5,692,193) (hereinafter Jagannathan) in view of Pinter et al. (USPN 6,457,023) (hereinafter Pinter).**

5. As per claim 1, Jagannathan teaches the invention as claimed, including a computer program product encoding a computer program for executing on a computer system a computer implemented method for managing allocation of program data in a target program between one

or more thread-specific heaps and at least one shared heap, the program data including thread-specific data and shared data, the computer implemented method comprising:

analyzing the target program during code compilation to distinguish between the thread-specific data of a first program thread and the shared data (col. 10 lines 21-35; col. 21 lines 55-57);

configuring the target program to allocate the thread-specific data of the first program thread to a first thread-specific heap, responsive to the analyzing operation (col. 20 line 56 - col. 21 line 26); and

configuring the target program to allocate the shared data to the shared heap, responsive to the analyzing operation (col. 21 lines 27-57).

6. Pinter teaches the invention as claimed, including analyzing the target program during code compilation (col. 1 line 65 - col. 2 line 6) to identify proven thread-specific data (col. 4 lines 36-43; col. 5 lines 5-18; col. 8 lines 42-49; col. 9 lines 21-31).

7. It would have been obvious to one of ordinary skill in the art to combine Jagannathan and Pinter because generational garbage collectors suffer from runtime overhead due to an inability to identify the lifetime of objects (Jagannathan, col. 20 lines 56-61). This deficiency, which is noted by Applicant, is also noted by Jagannathan. Pinter seeks to remedy the problems associated with generational garbage collectors by providing an estimate of object lifetime by performing pointer analysis. The pointer analysis also performs reachability analysis to identify if an object is accessed by a single thread or multiple threads (col. 2 lines 58-65). This pointer analysis proves that a particular object is accessed by a single thread, therefore being permissible to allocate on a local heap (col. 1 lines 7-11). Pinter points out that such data flow analysis can

be used to optimize compilers (col. 1 line 65 - col. 2 line 6), while Jagannathan allows for identification of shared and local data to be performed by compile-time analysis (col. 21 lines 55-57).

8. As per claim 2, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the analyzing operation comprises analyzing the target program to distinguish among the thread-specific data of the first program thread, the thread-specific data of a second program thread, and the shared data (col. 20 line 56 - col. 21 line 57), and wherein the computer implemented method further comprises:

configuring the target program to allocate the thread-specific data of the second program thread to a second thread-specific heap, responsive to the analyzing operation (col. 20 line 56 - col. 21 line 26).

9. As per claim 3, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the analyzing operation comprises:

identifying program data in the target program as the thread-specific data of the first program thread, if the program data is not referenced by any other program thread of the target program (col. 21 lines 7-26).

10. As per claim 4, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the analyzing operation comprises:

identifying program data in the target program as the thread-specific data of the first program thread based on a thread escape analysis (col. 21 lines 38-57).

11. As per claim 5, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the target program further includes a second program thread and the analyzing operation comprises:

identifying program data in the target program as the shared data, if the program data is referenced by the first program thread and the second program thread of the target program (col. 21 lines 27-37).

12. As per claim 6, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the analyzing operation occurs prior to the execution of the target program (col. 10 lines 21-35).

13. As per claim 15, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the thread-specific data occurs prior to execution of the target program (col. 14 lines 5-53).

14. As per claim 16, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the shared data occurs prior to execution of the target program (col. 14 lines 5-53).

15. As per claim 17, Jagannathan teaches the invention as claimed, including the computer program of claim 1 wherein the computer implemented method further comprises:

garbage collecting the thread-specific data from the first thread-specific heap independently of garbage collection of the shared data in the shared heap (col. 21 lines 7-26).

16. As per claim 18, Jagannathan teaches the invention as claimed, including the computer program of claim 1 wherein the computer implemented method further comprises:

garbage collecting the thread-specific data from the first thread-specific heap independently of garbage collection of a second thread-specific heap (col. 21 lines 7-26).

17. As per claim 19, Jagannathan teaches the invention as claimed, including the computer program of claim 1 wherein the computer implemented method further comprises:

garbage collecting the thread-specific data from the first thread-specific heap independently of the execution of another program thread in the target program (col. 21 lines 7-26).

18. As per claim 20, Jagannathan teaches the invention as claimed, including the computer program of claim 1 wherein the computer implemented method further comprises:

garbage collecting the shared data from the shared heap independently of garbage collection of the thread-specific data in the first thread-specific heap (col. 21 line 66 - col. 22 line 20).

19. As per claim 21, Jagannathan teaches the invention as claimed, including the computer program of claim 1 wherein the computer implemented method further comprises:

maintaining a remembered set identifying references to one or more shared data in the shared heap (col. 21 line 66 - col. 22 line 20); and

collecting the shared heap independently of garbage collection of the first thread-specific heap (col. 21 line 66 - col. 22 line 20).

20. As per claim 22, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the computer implemented method further comprises:

collecting a portion of the shared data from the shared heap to leave an uncollected portion of the shared data in the shared heap, the uncollected portion of the shared data including shared data that is referenced by thread-specific data of the first thread-specific heap that has not yet been scanned (col. 22 lines 12-20);

scanning the thread-specific data from the first thread-specific heap, responsive to the operation of collecting a portion of the shared data (col. 22 lines 12-20); and

collecting the uncollected portion of the shared data from the shared heap, responsive to the scanning operation (col. 22 lines 12-20).

21. As per claim 23, Jagannathan teaches the invention as claimed, including the computer program product of claim 22 wherein the computer implemented method further comprises:

collecting the thread-specific data from the first thread-specific heap, responsive to the operation of collecting a portion of the shared data (col. 21 lines 38-57).

22. As per claim 24, Jagannathan teaches the invention as claimed, including the computer program product of claim 1 wherein the shared heap is shared by a subset of the program threads of the target program (col. 12 line 66 - col. 13 line 22), wherein the subset of program threads includes less than all of the program threads of the target program (col. 12 line 66 - col. 13 line 22).

23. As per claim 25, Jagannathan teaches the invention as claimed, including a method of allocating of program data in a target program between one or more thread-specific heaps and at least one shared heap, the program data including thread-specific data and shared data, the method comprising:

analyzing the target program during code compilation to distinguish between the thread-specific data of a first program thread and the shared data (col. 10 lines 21-35; col. 21 lines 55-57);

configuring the target program to allocate the thread-specific data of the first program thread to a first thread-specific heap, responsive to the analyzing operation (col. 20 line 56 - col. 21 line 26); and

configuring the target program to allocate the shared data to the shared heap, responsive to the analyzing operation (col. 21 lines 27-57).

24. Pinter teaches the invention as claimed, including analyzing the target program during code compilation (col. 1 line 65 - col. 2 line 6) to identify proven thread-specific data (col. 4 lines 36-43; col. 5 lines 5-18; col. 8 lines 42-49; col. 9 lines 21-31).

25. As per claim 26, Jagannathan teaches the invention as claimed, including the method of claim 25 wherein target program further includes a second program thread and the analyzing operation comprises:

identifying program data in the target program as the shared data, if the program data is referenced by the first program thread and the second program thread of the target program (col. 21 lines 27-37).

26. As per claim 31, Jagannathan teaches the invention as claimed, including the method of claim 25 further comprising:

collecting a portion of the shared data from the shared heap to leave an uncollected portion of the shared data in the shared heap, the uncollected portion of the shared data including shared data that is referenced by thread-specific data of the first thread-specific heap that has not yet been scanned (col. 22 lines 12-20);

scanning the thread-specific data from the first thread-specific heap, responsive to the operation of collecting a portion of the shared data (col. 22 lines 12-20); and

collecting the uncollected portion of the shared data from the shared heap, responsive to the scanning operation (col. 22 lines 12-20).

27. As per claim 32, Jagannathan teaches the invention as claimed, including the method of claim 31 further comprising:

collecting the thread-specific data from the first thread-specific heap, responsive to the operation of collecting a portion of the shared data (col. 21 lines 38-57).

28. As per claim 34, Jagannathan teaches the invention as claimed, including a compiler for managing allocation of program data of a target program between a shared heap and a thread-specific heap, the program data including thread-specific data and shared data, the compiler comprising:

a program analyzer analyzing the target program during code compilation to distinguish between the thread-specific data of a first program thread and the shared data (col. 10 lines 21-35; col. 21 lines 55-57); and

a code specializer configuring the target program to allocate the thread-specific data of the first program thread to a first thread-specific heap (col. 20 line 56 - col. 21 line 26) and configuring the target program to allocate the shared data to the shared heap, responsive to the analyzing operation (col. 21 lines 27-57).

29. Pinter teaches the invention as claimed, including a program analyzer analyzing the target program during code compilation (col. 1 line 65 - col. 2 line 6) to identify proven thread-specific data (col. 4 lines 36-43; col. 5 lines 5-18; col. 8 lines 42-49; col. 9 lines 21-31).

30. As per claim 35, Jagannathan teaches the invention as claimed, including a computer program product encoding a computer program for executing on a computer system a computer implemented method for managing memory used for program data in a target program having

one or more thread-specific heaps and at least one shared heap, the program data including thread-specific data and shared data, the computer implemented method comprising:

analyzing the target program during code compilation to distinguish between the thread-specific data of a first program and the shared data (col. 10 lines 21-35; col. 21 lines 55-57);

allocating during target program code compilation thread-specific data associated with a first program thread of the target program to a first thread-specific heap, the thread-specific data being determined to be reachable only by the first thread (col. 20 line 56 - col. 21 line 26; col. 21 lines 55-57); and

allocating during target program code compilation the shared data to the shared heap, the shared data being deemed potentially reachable by a plurality of the program threads of the target program (col. 21 lines 27-57).

31. Pinter teaches the invention as claimed, including analyzing the target program during code compilation (col. 1 line 65 - col. 2 line 6) to identify proven thread-specific data (col. 4 lines 36-43; col. 5 lines 5-18; col. 8 lines 42-49; col. 9 lines 21-31).

32. As per claim 36, Jagannathan teaches the invention as claimed, including the computer program of claim 35 wherein the computer implemented method further comprises:

garbage collecting the thread-specific data from the first thread-specific heap independently of garbage collection of the shared data in the shared heap (col. 21 lines 7-26).

33. As per claim 37, Jagannathan teaches the invention as claimed, including the computer program of claim 35 wherein the computer implemented method further comprises:

garbage collecting the thread-specific data from the first thread-specific heap independently of the execution of another program thread in the target program (col. 21 lines 7-26).

34. As per claim 38, Jagannathan teaches the invention as claimed, including the computer program of claim 35 wherein the computer implemented method further comprises:

garbage collecting the shared data from the shared heap independently of garbage collection of the thread-specific data in the first thread-specific heap (col. 21 line 66 - col. 22 line 20).

35. As per claim 39, Jagannathan teaches the invention as claimed, including the computer program of claim 35 wherein the computer implemented method further comprises:

maintaining a remembered set identifying references to one or more shared data in the shared heap (col. 21 line 66 - col. 22 line 20); and

collecting the shared heap independently of garbage collection of the first thread-specific heap, based on the references identified in the remembered set (col. 21 line 66 - col. 22 line 20).

36. As per claim 40, Jagannathan teaches the invention as claimed, including a method of managing memory used for program data in a target program having one or more thread-specific heaps and at least one shared heap, the program data including thread-specific data and shared data, the method comprising:

analyzing the target program during code compilation to distinguish between the thread-specific data of a first program and the shared data (col. 10 lines 21-35; col. 21 lines 55-57);

allocating thread-specific data associated with a first program thread of the target program during code compilation to a first thread-specific heap, the thread-specific data being determined to be reachable only by the first thread (col. 20 line 56 - col. 21 line 26; col. 21 lines 55-57); and

allocating the shared data to the shared heap during code compilation, the shared data being deemed potentially reachable by a plurality of the program threads of the target program (col. 21 lines 27-57).

37. Pinter teaches the invention as claimed, including analyzing the target program during code compilation (col. 1 line 65 - col. 2 line 6) to identify proven thread-specific data (col. 4 lines 36-43; col. 5 lines 5-18; col. 8 lines 42-49; col. 9 lines 21-31).

38. As per claim 41, Jagannathan teaches the invention as claimed, including the method of claim 40 further comprising:

garbage collecting the thread-specific data from the first thread-specific heap independently of garbage collection of the shared data in the shared heap (col. 21 lines 7-26).

39. As per claim 42, Jagannathan teaches the invention as claimed, including the method of claim 40 further comprising:

garbage collecting the thread-specific data from the first thread-specific heap independently of the execution of another program thread in the target program (col. 21 lines 7-26).

40. As per claim 43, Jagannathan teaches the invention as claimed, including the method of claim 40 further comprising:

garbage collecting the shared data from the shared heap independently of garbage collection of the thread-specific data in the first thread-specific heap (col. 21 line 66 - col. 22 line 20).

41. As per claim 44, Jagannathan teaches the invention as claimed, including the method of claim 40 further comprising:

maintaining a remembered set identifying, references to one or more shared data in the shared heap (col. 21 line 66 - col. 22 line 20); and

collecting the shared heap independently of garbage collection of the first thread-specific heap, based on the remembered set (col. 21 line 66 - col. 22 line 20).

42. As per claim 45, Jagannathan teaches the invention as claimed, including the method of claim 40 further comprising:

collecting a portion of the shared data from the shared heap to leave an uncollected portion of the shared data in the shared heap, the uncollected portion of the shared data including

shared data that is referenced by thread-specific data of the first thread-specific heap that has not yet been scanned (col. 22 lines 12-20);

scanning the thread-specific data from the first thread-specific heap, responsive to the operation of collecting a portion of the shared data (col. 22 lines 12-20); and

collecting the uncollected portion of the shared data from the shared heap, responsive to the scanning operation (col. 22 lines 12-20).

43. As per claim 46, Jagannathan teaches the invention as claimed, including the method of claim 45 further comprising:

collecting the thread-specific data from the first thread-specific heap, responsive to the operation of collecting a portion of the shared data (col. 21 lines 38-57).

44. As per claim 47, Jagannathan teaches the invention as claimed, including a memory manager for managing heap memory in a computer system, the heap memory being used to store program data, the program data including thread-specific data and shared data, the memory manager comprising:

a program analyzer analyzing the target program during code compilation to distinguish between the thread-specific data of a first program and the shared data (col. 10 lines 21-35; col. 21 lines 55-57); and

an allocation module allocating thread-specific data associated with the first program thread of the target program to a first thread-specific heap, the thread-specific data being determined to be reachable only by the first thread (col. 20 line 56 - col. 21 line 26), and

allocating the shared data to the shared heap, the shared data being deemed potentially reachable by a plurality of the program threads of the target program (col. 21 lines 27-57).

45. Pinter teaches the invention as claimed, including a program analyzer analyzing the target program during code compilation (col. 1 line 65 - col. 2 line 6) to identify proven thread-specific data (col. 4 lines 36-43; col. 5 lines 5-18; col. 8 lines 42-49; col. 9 lines 21-31).

46. As per claim 48, Jagannathan teaches the invention as claimed, including the memory manager of claim 47 further comprising:

a garbage collector reclaiming memory associated with the thread-specific data from the first thread-specific heap independently of garbage collection of the shared data in the shared heap (col. 21 lines 7-26).

47. As per claim 49, Jagannathan teaches the invention as claimed, including the memory manager of claim 47 further comprising:

a garbage collector reclaiming memory associated with the thread-specific data from the first thread-specific heap independently of the execution of another program thread in the target program (col. 21 lines 7-26).

48. As per claim 50, Jagannathan teaches the invention as claimed, including the memory manager of claim 47 further comprising:

a garbage collector reclaiming memory associated with the shared data from the shared heap independently of garbage collection of the thread-specific data in the first thread-specific heap (col. 21 line 66 - col. 22 line 20).

49. As per claim 51, Jagannathan teaches the invention as claimed, including the memory manager of claim 47 wherein the memory manager maintains a remembered set identifying references to one or more shared data in the shared heap (col. 21 line 66 - col. 22 line 20) and further comprising:

a garbage collector reclaiming memory associated with the shared heap independently of garbage collection of the first thread-specific heap, based on the remembered set (col. 21 line 66 - col. 22 line 20).

50. **Claims 7-14, 27-30, and 33 are rejected under 35 U.S.C. 103(a) as being unpatentable over Jagannathan in view of Pinter in view of Benayon et al. (USPN 5,809,554) (hereinafter Benayon).**

51. As per claim 7, Benayon teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the thread-specific data comprises:

replacing an original allocation instruction in the target program with a new instruction that allocates the thread-specific data of the first program thread to the first thread-specific heap associated with the first program thread (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

52. It would have been obvious to one of ordinary skill in the art to combine Jagannathan, Pinter, and Benayon since dynamic changes in an execution context may require different allocation parameters depending on current conditions. Jagannathan is limited in this respect since the programming environment disclosed, Sting, is built on top of a sequential programming language, Scheme, that compiles and executes without modification (col. 10 lines 21-35). Although source level modifications to code cannot be made, Benayon provides a way of transparently modifying the allocation parameters of a thread such that heap allocation can be controlled if a user desires. After the resources are allocated for a specific thread, the allocation parameters return to the default. This achieves the claimed result of providing control over allocating data to the thread-specific heap or shared heap in a manner that maintains the integrity of the original source code.

53. As per claim 8, Benayon teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the thread-specific data comprises:

leaving an original allocation instruction in the target program to allocate the thread-specific data of the first program thread to the first thread-specific heap associated with the first program thread (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

54. As per claim 9, Benayon teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the shared data comprises:

leaving an original allocation instruction in the target program to allocate the shared data to the shared heap (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

55. As per claim 10, Benayon teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the shared data comprises:

replacing an original allocation instruction in the target program with a new instruction that allocates the shared data to the shared heap (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

56. As per claim 11, Benayon teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the thread-specific data comprises:

configuring an allocation parameter associated with the thread-specific data indicating that the thread-specific data of the first program thread is to be allocated in the one of the thread-specific heaps (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

57. As per claim 12, Benayon teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the thread-specific data further comprises:

allocating the thread-specific data of the first program thread to the first thread-specific heap associated with the first program thread, responsive to an allocation parameter (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

58. As per claim 13, Benayon teaches the invention as claimed, including the computer program product of claim 1 wherein the operation of configuring the target program to allocate the shared data comprises:

configuring an allocation parameter associated with the shared data indicating that the shared data is to be allocated in the shared heap (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

59. As per claim 14, Benayon teaches the invention as claimed, including the computer program product of claim 13 wherein the operation of configuring the target program to allocate the shared data further comprises:

allocating the shared data to the shared heap, responsive to the allocation parameter (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

60. As per claim 27, Benayon teaches the invention as claimed, including the method of claim 25 wherein the operation of configuring the target program to allocate the thread-specific data comprises:

replacing an original allocation instruction in the target program with a new instruction that allocates the thread-specific data of the first program thread to the first thread-specific heap associated with the first program thread (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

61. As per claim 28, Benayon teaches the invention as claimed, including the method of claim 25 wherein the operation of configuring the target program to allocate the thread-specific data comprises:

leaving an original allocation instruction in the target program to allocate the thread-specific data of the first program thread to the first thread-specific heap associated with the first program thread (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

62. As per claim 29, Benayon teaches the invention as claimed, including the method of claim 25 wherein the operation of configuring the target program to allocate the shared data comprises:

replacing an original allocation instruction in the target program with a new instruction that allocates the shared data to the shared heap (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

63. As per claim 30, Benayon teaches the invention as claimed, including the method of claim 25 wherein the operation of configuring the target program to allocate the shared data comprises:

leaving an original allocation instruction in the target program to allocate the shared data to the shared heap (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

64. As per claim 33, Benayon teaches the invention as claimed, including the method of claim 25 wherein the operation of configuring the target program to allocate the thread-specific data comprises:

configuring an allocation parameter associated with the thread-specific data indicating that the thread-specific data of the first program thread is to be allocated in the first thread-specific heap (col. 2 lines 53-58; col. 9 line 25 - col. 10 line 29).

Response to Arguments

65. **Applicant's arguments with respect to claims 1-51 have been considered but are moot in view of the new grounds of rejection.**

Conclusion

66. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Syed J Ali whose telephone number is (571) 272-3769. The examiner can normally be reached on Mon-Fri 8-5:30, 2nd Friday off.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai T An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Syed Ali
January 24, 2005

MENG-AL T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100